UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/944,685	08/31/2001	Henry Fang	SLA 1070	2111

7590 08/26/2005

David C. Ripma
Patent Counsel
Sharp Laboratories of America, Inc.
5750 NW Pacific Rim Boulevard
Camas, WA 98607

EXAMINER	
VO, TED T	

ART UNIT	PAPER NUMBER
2192	

DATE MAILED: 08/26/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/944,685	FANG, HENRY	
	Examiner	Art Unit	
	Ted T. Vo	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *5/31/05 (Appeal Brief)*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-17* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-17* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the communication filed on 05/31/2005.

Claims 1-17 are pending in the application.

Response to Arguments

2. In view of the Appeal Brief filed on 05/31/2005, PROSECUTION IS HEREBY REOPENED. A new ground of rejection is set forth below.

To avoid abandonment of the application, appellant must exercise one of the following two options:

(1) file a reply under 37 CFR 1.111 (if this Office action is non-final) or a reply under 37 CFR 1.113 (if this Office action is final); or,

(2) request reinstatement of the appeal.

If reinstatement of the appeal is requested, such request must be accompanied by a supplemental appeal brief, but no new amendments, affidavits (37 CFR 1.130, 1.131 or 1.132) or other evidence are permitted. See 37 CFR 1.193(b)(2).

- Response to Applicants' arguments to Claim 1-17, rejected under 35 U.S.C. 112 second paragraph is withdrawn.

Applicants are respectfully suggested to view the following definitions:

> JNI (Java Native Interface) A Java programming interface, or API, that allows developers to access the languages of a host system and determine the way Java integrates with native code. The JNI has been a point of contention between Sun and Microsoft, since Microsoft seeks to create its own native code interface and Sun claims this violates their licensing agreement. http://www.webopedia.com/TERM/J/JNI.html

JNI: Java Native Interface is a standard programming interface for writing Java native methods and embedding the Java virtual machine into native applications. The primary goal is binary compatibility of native method libraries across all Java virtual machine implementations on a given platform.
JDK 1.2 extends the Java Native Interface (JNI) to incorporate new features in the Java platform. The changes are driven by licensee and user comments.
You must declare all methods, whether Java programming language methods or native methods, within a Java programming language class. When you write a method implementation in a programming language other than the Java programming language, you must include the keyword **native** as part of the method's definition within the Java programming language class. The **native keyword** signals to the Java compiler that the function is a native language function.
http://cities.lk.net/JavaDictionary2.html

FAV -- FAV is an acronym for Full Audio Visual device. An FAV contains a complete set of the software elements comprising the HAVi Architecture. This device class generally has a rich set of resources and is capable of supporting a complex software environment. The primary distinguishing feature of an FAV is the presence of a runtime environment for Java bytecode. This allows an FAV to upload bytecode from other devices and so provide enhanced capabilities for their control. Likely candidates for FAV devices are Set-top Boxes, DTV receivers, general purpose home control devices, residential gateways, and home PCs.

IAV -- IAV is an acronym for Intermediate Audio Visual device. An IAV is generally lower in cost than an FAV and more limited in resources. IAVs do not provide a runtime environment for Java bytecode and so cannot act as controllers for arbitrary devices within the home network. However, an IAV may provide native support for control of particular devices on the home network.

http://www.havi.org/techinfo/faq/faq_device.html#combo

All the above definitions show that any use of JNI for accessing a Java native code that stored in a storage is only to conform to what it is provided by SUN's elements. The elements such as JAR, ResourcBundle, Virtual Key, API, JNI, are belonged to the standard code formats developed by SUN. When SUN develops a Java code store that store Java archive files, SUN should know how to access every element in a JAR file stored in such a Java code store. Otherwise, SUN cannot move on to do with its development.

Specification note: All the descriptions in the specification shows that accessing and retrieval in using a HAVi Level 2 UI interface are step-by-step done by a user. The plain language of Claims 1, 7, and 12 clearly are step-by-step done by a user who saty infront of HAVi framework. It should be noted a manipulation disclosed by a specification with a requirement of a person would not be an act of patentability.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

4. Claim 1-6 are rejected under 35 U.S.C. 102(a) as being anticipated by HAVi SPECIFICATION Version 1.1, 5-2001 (hereinafter: HAVi).

Given the broadest reasonable interpretation of followed claims in light of the specification.

As per claim 1: HAVi discloses,

In a signal bearing medium tangibly embodying a program of machine-readable instructions executed by an open source, interoperable IEEE 1394 specification digital device, a method for defining device user interface controls, the method comprising:

from a level two (L2) graphical user interface (GUI), accessing a JAR file; (See page 5, row 10, 'Level 2 interoperability', in page 22, third paragraph, 'access device functionality', page 18, section 2.5.2, Level 2 UI, page 24, section 2.9.2, 'are JAR files'); *and,*

in response to accessing the JAR file, retrieving virtual key information (See page 24, section 2.9.2, 'obtained from "code unit", page 429, sec. 8.3.2.4 and 8.3.2.5, see: User Input Representation, 'getString', 'getColor', 'getSymbol', 'VK_COLORED_KEY_0',,, 'VK_COLORED_KEY_5', *retrieving virtual key information*).

As per claim 2: HAVi discloses, *The method of claim 1 wherein accessing a JAR file includes accessing a JAR file stored in read only memory (ROM)* (See page 455, section 9.5,Table 18, HAVi Configuration ROM Requirement).

As per claim 3: HAVi discloses, *The method of claim 2 wherein retrieving virtual key information includes retrieving virtual key information from a JAR file model selected from the group including static classes and data arrays.* (See page 90, section 3.10.2, Figure 26) by showing a code unit of a Jar file that includes *static classes and data arrays.*

As per claim 4: HAVi discloses, The *method of claim 3 wherein retrieving virtual key information in response to accessing the JAR file includes retrieving an application bundled with the virtual key information* (See page 429, section 8.3.2.5, User Input Representation, "by calling "getString", "getColor", "getSymbol", *bundled with the virtual key information*).

As per claim 5: HAVi discloses, *The method of claim 4 in which a first microprocessor machine using a first operating system is included; the method further comprising: receiving virtual key information as Java source code; using a Java compiler, compiling the Java source code into Java virtual machine (JVM) byte codes for the first operating system; and, using jar tools, archiving the JVM byte codes into a JAR file stored in ROM* (See Page 395, Java 1.1 Core API, and Jave.lang.Compiler).

As per claim 6: HAVi discloses, The *method of claim 5 further comprising: receiving the application as Java source code; using a Java compiler, compiling the Java source code into Java virtual machine (JVM) byte codes for the first operating system; and, using jar tools, archiving the JVM byte codes into a JAR file stored in ROM* (See Page 395, Java 1.1 Core API, and Jave.lang.Compiler).

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 7-11 are rejected under 35 U.S.C. 103(a) as being unpatentable over HAVi SPECIFICATION Version 1.1, 5-2001 in view of Jordan, "Java in the Home: OSGi Residential Gateways", 7-2000.

As per claim 7: HAVi discloses,

In a signal bearing medium tangibly embodying a program of machine-readable instructions executed by an open source, interoperable IEEE 1394 specification digital device, a method for defining device user interface controls, the method comprising:

from a level two (L2) graphical user interface (GUI) accessing a Java input/output (I/O) ResourceBundle ; and,

in response to accessing the ResourceBundle, retrieving virtual key information.

HAVi, discloses a specification that includes a level 2 UI which is used to access into Java files, called code units.

From this level 2 (Chapter 8), the I/O map of a code unit (Figure 26), which is Jar file, is accessed using Java API (Chapter 8: sec 8.1).

Using "Java events" such as getString, getColor, or getSymbol, (Chapter 8: sec. 8.3.2.4, 8.3.2.5) the virtual key information such as VK_COLORED_KEY_X, (X=1, 2, 3, ...), are retrieved via Jar files.

HAVi does not clearly address *accessing a Java input/output (I/O) ResourceBundle.*

However, Figure 25 and 26 show these JAR files (code units) are accessed via I/O of a Jar file.

Jordan discloses "Java in the Home: OSGi Residential Gateways", that shows Jar file is a standard used in HAVi (JavaReport 39, the fourth column of the page), where a JAR file contains Java resource bundles indicating where resource information resided (See JavaReport 41-42: "Bundles").

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the disclosures of HAVi specification 1.1 and the disclosure of Jar files as related to standard software used in HAVi by Jordan because I/O Resource bundles is conforming to JAR file format where the resource information is resided.

Claims 8-11 are further limitations of Claim 7, where the teaching of HAVi is further extended to Resource Bundles mentioned by Jordan. It is obvious to a person of ordinary skill as being conforming to a standard of Jar file format where resource information is defined in a Jar file.

As per claim 8: HAVi discloses, *The method of claim 7 wherein accessing the ResourceBundle includes using a ResourceBundle application program interface (API) to specify a property file* (Referring to Level 2 UI, page 425; where Level 2 UI is related to API, and see page 429, section 8.3.2.5, User Input Representation, referring to class that defines an event having a known representation; For example six colored key events).

As per claim 9: HAVi discloses, *The method of claim 8 in which a first microprocessor machine using a first operating system is included; the method further comprising: maintaining an application in a protocol associated with the first operating system; and, wherein accessing the ResourceBundle includes using a ResourceBundle API to specify a property file stored in a file system associated with the first microprocessor machine* (See pages 414-419, section 7.4 Code Units (for install and uninstall Jar files));

As per claim 10: HAVi discloses, *The method of claim 9 wherein using a ResourceBundle API to specify a property file stored in the file system includes specifying a property file stored in an input/output (I/O) device selected from the group of storage devices including hard disks and Flash memory* (See page 6, section 1.4 referring to "persistent memory" for storage devices).

As per claim 11: HAVi discloses, *The method of claim 10 further comprising: receiving virtual key information as text-based properties attributes in a ResourceBundle property file; integrating the virtual key information into a table of virtual key characteristics; and, storing the virtual key characteristics table as the ResourceBundle property file* (See page 429, section 8.3.2.5, User Input Representation, referring to the representation getString that allows a text representation).

7. Claims 12-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over HAVi SPECIFICATION Version 1.1, 5-2001 in view of Sun, "Java native interface" (1995-2000).

As per claim 12: HAVi discloses,

In a signal bearing medium tangibly embodying a program of machine-readable instructions executed by an open source, interoperable IEEE 1394 specification digital device, a method for defining device user interface controls, the method comprising:

from a level two (L2) graphical user interface (GUI) calling a Java native interface (JNI); at the JNI, using Java byte codes to call a storage driver from the storage driver, accessing a mapped memory; and,

in response to accessing the mapped memory, retrieving virtual key information

HAVi, discloses a specification that includes a level 2 UI which is used to access into Java files, called code units.

From this level 2 (Chapter 8), code units (Figure 26) which are Jar files, are accessed using API and subsets of AWT (Chapter 8: sec 8.1; Table 15).

In accordance to the Figure 25 and 26, HAVi shows the maps of storage drivers (Figure 25), where code units are resided. Using "Java events" (has equivalent functionality: *Java byte codes*) such as getString, getColor, or getSymbol, which are Java code (Chapter 8: sec. 8.3.2.4, 8.3.2.5) the virtual key informations such as VK_COLORED_KEY_X, (X=1, 2, 3, ...), are retrieved.

HAVi does not clearly address *calling a Java native interface (JNI); at the JNI, using Java byte codes to call a storage driver.*

However, Figure 25 and 26 show these code units are connectively driven via the Level 2 UI, and such accessibility is used Java elements such as AWT, Java API, etc.

Sun discloses Java native interface (http://java.sun.com/j2se/1.3/docs/guide/jni/index.html) (1999-2000) shows JNI in "The AWT Native Interface"

(http://java.sun.com/j2se/1.3/docs/guide/awt/AWT_Native_Interface.html)

is required for accessed to Java code units, as it is noted

> "Standard Edition includes JNI, the Java Native Interface. Most Java developers will never need to use it, but the interface is invaluable in certain situations because it provides the only way for Java byte code to interact directly with application code that has been compiled to the native machine instructions for the host microprocessor. JNI is most often used as an ``escape valve" to enable access to platform functionality not yet supported by the Java programming language. For

example, you could use JNI to integrate with native code that communicates with a peripheral device, such as a scanner, connected to a system via a USB port."

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the disclosures of HAVi specification 1.1 and Java Native Interface of Sun because calling a Java native interface, by using Java byte codes to access Java native code is conforming to a standard designed by SUN for accessing to a Java code unit.

Claims 13-17 are further limitations of Claim 12, where the teaching of HAVi is further extended to Java Native interface of SUN. It is obvious to a person of ordinary skill as being conforming to a standard of Java interface designed by Sun for accessing its native code.

As per claim 13: *The method of claim 12 wherein accessing a mapped memory includes accessing a mapped memory stored in an electrically erasable programmable read only memory (EEPROM)* (See page 55, Figure 18, code unit; and see page 455, section 9.5,Table 18, HAVi Configuration ROM Requirement).

As per claim 14: *The method of claim 13 wherein retrieving virtual key information includes retrieving virtual key information from mapped memory in a binary format* (See section 9.11.1, pages 429-430, for Keys that represents virtual key information in numbers).

As per claim 15: *The method of claim 14 wherein using Java byte codes to call a storage driver at the JNI includes converting the Java byte code to binary format addresses (inherent in JVM); and, wherein accessing a mapped memory from the storage driver includes using the binary format addresses to access ASCII codes stored in the EEPROM* (See section 9.11.1, pages 429-430, for Keys that represents virtual key information in numbers).

As per claim 16: *The method of claim 15 in which a first microprocessor using a first operating system is included; the method further comprising: receiving the storage driver as first operating system machine codes; and, storing the storage driver as machine code* (See page 54, last two paragraphs, see page 55, Figure 18, and third paragraph).

As per claim 17: *The method of claim 16 further comprising: receiving virtual key information as binary format code; using the storage driver, cross-referencing the virtual key information with EEPROM addresses; and, storing the virtual key information in the EEPROM as machine code* (See page 54, last two paragraphs, see page 55, Figure 18, and third paragraph).

Conclusion

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The examiner can normally be reached on 8:00AM to 5:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3694.

The facsimile number for the organization where this application or proceeding is assigned is the Central Facsimile number **571-273-8300**.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Ted T. Vo
Primary Examiner
Art Unit 2192
August 16, 2005